CIS 5500 Project Proposal: Funding-Aware Market Maker

Fall 2025

Team

- **Names:** Gaurav Malhotra, Albert Opher, Ishaan Shah, Madhav Sharma
- **Emails:** gmal2005@seas.upenn.edu, albert.w.opher.iv@gmail.com, ishaan8@wharton.upenn.edu, madhavsh@wharton.upenn.edu
- **GitHub:** GauravMalhotra010, Albinator3000, IshaanShah25082005, Madhav-Sharma-07

Idea: We will build a web app that helps a market maker adjust quote widths and inventory targets around perpetual futures funding events. The system ingests large historical datasets of minute bars, funding rates, and open interest for dozens of symbols, computes event windows (pre/post funding), and surfaces timing/extent of expected adverse selection and fill opportunity. Users can interactively select symbols and time spans, visualize effects, and view pre/post optimization timings for complex SQL queries.

Datasets, Scale, and Access We will use *two or more large, overlapping sources* keyed by `(symbol, timestamp)`. All are public and scriptable.

Dataset	Provider	Granularity / Notes	Link
USDT-Margined Futures **Klines**	Binance	1m candles (open, high, low, close, volume, taker buy volume, trade count). Monthly archives for all symbols. *Scale:* 1 symbol $\approx$ 3.1M rows over 6 years; 50+ symbols $>$ 150M rows.	data.binance.vision (UM klines)
Funding Rate History	Binance	8h funding prints per perpetual symbol; long history via REST; also monthly CSV archives.	API: GET /fapi/v1/fundingRate CSV archives
Open Interest (OI)	Binance	OI snapshots (API offers up to $\tilde{1}$ month history windows); sufficient for event windows; can roll daily.	Open Interest Statistics
Maybe: Premium Index Klines	Binance	Premium/discount vs index price; useful for basis.	PremiumIndexKlines

Scale: Minute bars for dozens of symbols yield hundreds of millions of rows; funding/OI add event labels and covariates. All datasets share `(symbol, timestamp)` for natural joins, enabling heavy event-study queries.

Overlap & Entity Resolution

- **Keys:** Normalize `symbol` (e.g., BTCUSDT, ETHUSDT) and timestamps to UTC minutes/seconds. Funding prints occur every 8 hours; align each funding event to a window $[t - \Delta, t + \Delta]$.
- **Cleaning:** Deduplicate symbol aliases, filter delisted/illiquid symbols, enforce trading hours for non-24/7 venues (N/A here). Check missing klines; backfill only metadata (not prices).
- **Integrity:** Foreign keys `funding(symbol, ts)` $\rightarrow$ `symbols(symbol)`; check constraints on price/volume ranges.

ER/Schematic Design (3NF/BCNF) We target a normalized core with materialized views for heavy windows.

- symbols(symbol, base, quote, onboard_date)
- klines(symbol, open_time, close_time, open, high, low, close, volume, taker_buy_base, taker_buy_quote, n_trades)
- funding(symbol, ts, rate) (8h cadence)
- open_interest(symbol, ts, oi)
- events(symbol, event_ts, kind) (populated from funding, releases)
- (Derived) minute_returns(symbol, ts, r1m, rv_30m, rv_1h)

PostgreSQL DDL stub:

```
CREATE TABLE symbols (
  symbol TEXT PRIMARY KEY,
  base   TEXT NOT NULL,
  quote  TEXT NOT NULL,
  onboard_date DATE
);

CREATE TABLE klines (
  symbol TEXT REFERENCES symbols(symbol),
  open_time TIMESTAMPTZ,
  close_time TIMESTAMPTZ,
  open NUMERIC, high NUMERIC, low NUMERIC, close NUMERIC,
  volume NUMERIC,
  taker_buy_base NUMERIC, taker_buy_quote NUMERIC,
  n_trades INT,
  PRIMARY KEY (symbol, open_time)
) PARTITION BY RANGE (open_time);

CREATE TABLE funding (
  symbol TEXT REFERENCES symbols(symbol),
  ts TIMESTAMPTZ,
  rate NUMERIC,
  PRIMARY KEY (symbol, ts)
);

CREATE TABLE open_interest (
  symbol TEXT REFERENCES symbols(symbol),
  ts TIMESTAMPTZ,
  oi NUMERIC,
  PRIMARY KEY (symbol, ts)
);

-- Derived features
CREATE MATERIALIZED VIEW minute_returns AS
SELECT symbol,
       open_time AS ts,
       LN(close) - LN(LAG(close) OVER (PARTITION BY symbol ORDER BY open_time)) AS r1m,
```

```
       STDDEV_SAMP(LN(close) - LN(LAG(close) OVER (PARTITION BY symbol ORDER BY open_time))
         OVER (PARTITION BY symbol ORDER BY open_time ROWS BETWEEN 30 PRECEDING AND CURRENT
FROM klines;
```

Candidate Queries

1. **Event & Drift (*complex*).** For each funding event, compute cumulative abnormal return (CAR) in windows $[-60, +180]$ minutes vs. symbol's recent beta to BTC (CTEs, window joins, scalar subqueries).

2. **Rate-Decile Effects (*complex*).** Bucket funding rates into deciles by day; compute post-event 60-minute markouts and test monotonicity across deciles with `GROUPING SETS`.

3. **Funding × OI Regime (*complex*).** Identify windows where $|rate| > p90$ and OI is above its 14-day percentile; return the top-K symbols by mean post-event drift.

4. **"Never Negative" Screen (*complex*).** Symbols that *never* exhibit negative 30-minute CAR after funding when daily realized volatility is below median (`NOT EXISTS` universal quantification).

5. **Spread/Width Suggestion.** For a chosen symbol, recommend quote widening Δ when upcoming funding decile≥ 9 & current RV$> p75$; output timestamps and suggested widths.

6. **Seasonality/Hour-of-Day.** Average CAR by hour-of-day of funding vs. non-funding pseudo events.

7. **Cross-sectional Panel.** Join premium index klines to compute basis; does extreme basis amplify funding-event drift?

Optimization Plan (to meet $>$ 15s pre-opt $\rightarrow$ $<$ 1s post-opt)

- **Partitioning:** Range partitions on `klines.open_time` by month; funding/OI by week. Prune scans during event windows.
- **Indexes:** Composite `(symbol, open_time)` on klines; `(symbol, ts)` on funding/OI; partial indexes for active trading years.
- **Materialized Views:** (i) funding windows expanded to minute resolution; (ii) minute returns and rolling RV; (iii) per-symbol decile buckets. Refresh on load.
- **Join Strategy:** Ensure merge-join/bitmap via matching sort orders; pre-aggregate per minute before joining to 8h events.
- **Query Rewrites:** Replace nested subqueries with CTEs; push filters; use `DISTINCT ON` only where needed.
- **Explain/Analyze:** Capture pre/post plans and wall-clock, include in demo.

Web App (multi-page)

- **Event Study Explorer:** Select a symbol/date; plot CAR, OI, and funding decile bands.
- **Regime Screener:** Filter symbols by conditions (e.g., high |rate|, high OI) and list windows with statistics.
- **Quote-Width Lab:** Parameter slider for widening rules; show historical P&L proxy (spread capture minus drift).

- **Timings Dashboard:** For two complex queries, show pre/post optimization runtimes and query plans.

Feasibility, Risks, and Mitigations

- **Rate limits / data volume:** Use monthly CSV archives (klines) and funding CSVs first; REST for OI windows. Batch-ingest and compress (`COPY FROM`).
- **Clock alignment:** Funding at 8h cadence; we generate minute windows around each funding timestamp; verify timezone/UTC.
- **Symbol churn:** Maintain `symbols` table; ignore low-liquidity symbols below volume thresholds.
- **Legal/ToS:** Use only public endpoints/archives; cite providers; respect API limits. This is an academic, non-commercial project.

References (dataset docs)

- Binance USDT-Margined Futures Klines (archives): https://data.binance.vision/?prefix=data/futures/um/monthly/klines/
- Binance Funding Rate History (API): https://developers.binance.com/docs/derivatives/usds-margined-futures/market-data/rest-api/Get-Funding-Rate-History
- Binance Funding Rate (archives): https://data.binance.vision/?prefix=data/futures/um/monthly/fundingRate/
- Binance Open Interest Statistics (API): https://developers.binance.com/docs/derivatives/usds-margined-futures/market-data/rest-api/Open-Interest-Statistics
- Premium Index Kline Data (API): https://developers.binance.com/docs/derivatives/usds-margined-futures/market-data/rest-api/Premium-Index-Kline-Data

Note: We will keep the scope within the course limits (security, deployment optional) and focus on database design, heavy SQL, and optimization.